行列の四則演算

行列とその四則演算のための header ファイル mymat.h を用意した ()。このファイルでは行列を表すクラス型 myMat が宣言されているので、このクラス型を用いて行列演算を行うとリスト１に示すように、行列の四則演算を普通の数値の演算と同じように書くことができる。

リスト１　行列演算の例。

```
#include "stdafx.h"
#include "mymat.h"
#include <iostream>

using namespace System;
using namespace std;
using namespace myLib;

int main(array<System::String ^> ^args)
{
        myMat a(2,2);
        a[0][0] = 1.0;  a[0][1] = 2.0;
        a[1][0] = 3.0;  a[1][1] = 4.0;
        cout << "a =" << endl;
        WriteMat( 5, a );

        myMat b(2,2);
        b[0][0] = 10.0; b[0][1] = 20.0;
        b[1][0] = 30.0; b[1][1] = 40.0;
        cout << endl << "b =" << endl;
        WriteMat( 5, b );

        myMat sum(2,2);
        sum = a + b;
        cout << endl << "a + b =" << endl;
        WriteMat( 5, sum );

        myMat dif(2,2);
        dif = a - b;
        cout << endl << "a - b =" << endl;
        WriteMat( 5, dif );

        myMat J(2,2);
        J[0][0] = 0.0;  J[0][1] = 1.0;
        J[1][0] = 1.0;  J[1][1] = 0.0;
        cout << endl << "J =" << endl;
        WriteMat( 5, J );

        cout << endl << "J * a =" << endl;
        WriteMat( 5, J * a );
        cout << endl << "a * J =" << endl;
```

```
        WriteMat( 5, a * J );

        try {
                J[0][0] = a[9][9];
        }
        catch (exception e){
                cout << "exception..." << e.what() << endl;
        }

        Console::WriteLine();
        Console::WriteLine("Enterキーを押すと終了します");
        Console::ReadLine();
        return 0;
}
```

m × nの大きさの行列 a を用意するときは、

myMat　a(m, n);

と宣言すればよい。

a の(i,j)要素は

a[i][j]

で表わされる。

別の同じサイズの行列が

myMat　b(m, n);

と宣言されているとき、それらの和と差は

a + b　および　a – b

で表わされる。和あるいは差を

myMat　sum(m, n);

myMat　diff(m, n);

と宣言された行列に格納するときは代入演算子＝を用いて

sum = a + b;

diff = a – b;

と書けばよい。

行列の要素へのアクセスで、添え字が範囲外のものであるときは例外 exception が生成される。この例は、リスト１の最後の方の

```
        try {
                J[0][0] = a[9][9];
        }
        catch (exception e){
                cout << "exception..." << e.what() << endl;
        }
```

で示されている。

ヘッダーファイル mymat.h には、掛け算、割り算も用意されている。mymat.h ファイ

ルの内容をリスト２に示す。用意されている演算子は

[ ],　=,　+=,　−=,　*=,　/=,　+,　−,　*,　/

である。他に転置行列を返す関数 matTransp、行列の要素の値の標準出力を行う関数 WriteMat などを用意されており、これらを用いるとプログラムが簡潔になる。mymat.h に宣言されているものの定義は mymat.cpp を参照のこと。Visual C++で mymat.h をインクルードして利用するときは、mymat.cpp も同じフォルダに用意しておいて、メニュ「プロジェクト｜既存の項目の追加」によって mymat.cpp を追加しておく。この追加を行わないとリンク時にエラーとなる。

リスト１のプログラムの実行結果を、リスト２の後の図１に示す。

リスト２　mymat.h の内容。

```
#ifndef MyMat_CK
#define MyMat_CK

#include <iostream>

using namespace std;

#define MATDEBUG 1  //   1 -> exception on,  0 -> exception off.

namespace myLib {


class myVec {
        public:
                int vctrn;
                double * v;

                myVec(): vctrn(0), v(NULL) { }
                explicit myVec(int nn): vctrn(0), v(NULL) {
                        if (nn > 0) {
                                vctrn = nn;
                                v = new double[vctrn];
                                for (int i = 0; i < vctrn; i++) v[i] = 0.0;
                        }
                }
                myVec(const myVec & tmplt){
                        vctrn = tmplt.vctrn;

                        if (vctrn > 0){
                                v = new double[vctrn];
                                for (int i = 0; i < vctrn; i++)   v[i] = tmplt.v[i];
                        }
                        else v = NULL;
                }

                virtual ~myVec(){ if (v != NULL) delete[] v; }

                int set_vsize( int nn );
```

```
                double & operator[](int nn);

                myVec & operator = (const myVec & R);
};

class myMat {
        public:
                int m, n;
                myVec * vv;

                myMat(): m(0), n(0), vv(NULL){    }
                myMat(int mm, int nn): vv(NULL) {
                        if (!((mm > 0) && (nn > 0))){
                                m = 0; n = 0; vv = NULL;
                        } else {
                                set_size(mm, nn);
                        }
                }

                myMat(const myMat & tmplt){
                        m = tmplt.m;
                        n = tmplt.n;

                        if (m > 0){
                                vv = new myVec[m];
                                for (int i = 0;  i < m; i++){
                                        vv[i].v = new double[n];
                                        vv[i].vctrn = n;

                                        for (int j = 0; j < n; j++){
                                                vv[i].v[j] = tmplt.vv[i].v[j];
                                        }
                                }
                        }
                        else vv = NULL;
                }

                virtual ~myMat(){ if (vv != NULL) delete[] vv; }

                //     (Re-)size the matrix
                int set_size( int mm, int nn );


                int get_size_row(){ return m; }

                int get_size_col(){ return n; }

                //       *this[]
                myVec & operator[](int mm);

                //       *this = R
                myMat & operator = (const myMat & R);
```

```
                    //      *this = *this + R
                    myMat & operator += (const myMat & R);

                    //      *this = *this - R
                    myMat & operator -= (const myMat & R);

                    //      *this = *this * R
                    myMat & operator *= (const myMat & R);

                    //      *this = *this / R
                    myMat & operator /= (const myMat & R);
};

//    Calculate L + R
myMat operator+ (const myMat & L, const myMat & R);

//    Calculate L - R
myMat operator- (const myMat & L, const myMat & R);

//    Calculate L * R
myMat operator* (const myMat & L, const myMat & R);

//    Calculate L / R
myMat operator/ (const myMat & L, const myMat & R);

//      calculate a'
myMat matTransp( myMat & a );

//      myVecの書き出し
void WriteVec( myVec Vec );

//      myVecの書き出し：書き出し幅指定
void WriteVec( int w, myVec Vec );

//      myMatの書き出し
void WriteMat( myMat Mat );

//      myMatの書き出し：書き出し幅指定
void WriteMat( int w, myMat Mat );

//    c = a + b
int matAdd( myMat & a, myMat & b, myMat & c, int m, int n );


//    逆行列の計算：  掃き出し法
//     a : 逆行列を求める行列；    invMat : 逆行列を返す行列
int calcInvMat( myMat a, myMat & invMat, int n );


}   //   End of myLib

#endif
```

```
a =
     1     2
     3     4

b =
    10    20
    30    40

a + b =
    11    22
    33    44

a - b =
    -9   -18
   -27   -36

J =
     0     1
     1     0

J * a =
     3     4
     1     2

a * J =
     2     1
     4     3
exception...Index Range Over Error in class myMat...RowIndex = 9

Enter キーを押すと終了します
```

図１　リスト１の実行結果。